Available online at www.sciencedirect.com

SciVerse ScienceDirect

23rd International Meshing Roundtable 00 (2014) 000–000

Procedia Engineering

www.elsevier.com/locate/procedia

Towards automated hybrid-prismatic mesh generation

Maximilian Tomac[1], David Eller[1]

KTH Aeronautical and Vehicle Engineering, Stockholm, Sweden

Abstract

An open-source implementation of an efficient mesh generation procedure for hybrid prismatic-tetrahedral meshes intended for use in Reynolds-averaged Navier-Stokes solutions is presented. The method employed combines the established, and very fast, Delaunay-based tetrahedral mesh generator TetGen with a novel technique for the creation of a prismatic layer. Satisfactory mesh quality is demonstrated by comparing solutions obtained using the new meshes with reference data computed on advancing-front grids. Mesh generation time is shown to be substantially less than with some other methods. Overall, the presented implementation is deemed a valuable tool for cases where many meshes need to be generated for routine analyses and turnaround time is critical.

Peer-review under responsibility of organizing committee of the 23rd International Meshing Roundtable (IMR23).

1. Introduction

The creation of high-quality discretizations for use in viscous flow simulations remains a challenging task [1–3]. Even with modern software tools and substantial human effort, the application of state-of-the-art mesh generation algorithms in the presence of geometric features such as concave corners may still result in inadequate local mesh quality, which can severely affect the resolution of important flow features [4].

To address such issues, mesh generation tools for hybrid unstructured grids often expose a considerable number of algorithm configuration parameter [5–7], many of which have a profound influence on the robustness of the process. The resulting flexibility does indeed enable the creation of sufficiently resolved hybrid meshes, although parameter selection often requires a considerable effort even for an experienced user. In a production environment where a large number of detailed simulations of a single aircraft configuration are performed, the cost in terms of man-hours may be entirely acceptable. An example of such a situation would be the drag evaluation of an airliner. For other applications with requirements for short turn-around time, e.g. in aerodynamic load computations for many different geometric configurations, a more automated approach is desirable.

Since an automatic mesh generation procedure cannot rely on user intervention for the resolution of problems resulting from geometric complications, a robust strategy for the handling of surface geometry features encountered in realistic aircraft configurations must be implemented. One possible solution is to accept a hybrid mesh with known regions of low element quality, and then to augment the unstructured volume mesh with one or a number of structured overset blocks which can be manually adapted to better resolve critical geometric features [8,4]. Obviously, this

E-mail addresses: max.tomac@gmail.com (Maximilian Tomac)., dlr@kth.se (David Eller).

Peer-review under responsibility of organizing committee of the 23rd International Meshing Roundtable (IMR23).

approach requires support for overset grids, which is not necessarily available in all state-of-the-art flow solvers for industrial use. Another approach to improve robustness is the transition to a combination of prismatic, tetrahedral and octtree-based mesh generation procedures, which has been shown to allow a surprising flexibility in the presence of difficult geometric features [9]. From the information available at this time, it is however not clear how the resulting mixed mesh topology and locally biased edge alignment will affect the solution accuracy for three-dimensional boundary layer flows [10].

The approach presented here is based on a segregated prismatic/tetrahedral mesh generation procedure, and aims to achieve robustness by means of local geometric modifications. Criteria chosen and algorithmic modifications make use of similar principles as in earlier work [11–13], but are adapted for the specific requirements of mesh generation for aircraft configurations. An existing set of open-source tools is exploited for mesh data structures, file format support, surface mesh generation and the creation of tetrahedral volume meshes.

Surface mesh generation capabilities of the current tool-chain have been presented earlier [14]; therefore, the present paper is focused on the procedures employed in the volume mesh generation step. Such a decomposition is possible as the algorithms do not currently exploit any coupling with the surface mesh generation stage and can therefore also be utilized to create a hybrid prismatic-tetrahedral mesh around an existing triangular surface created by any other software, provided that this surface mesh is of sufficient quality. Requirements for surface mesh quality are discussed in Section 3.3.

2. Aim

In contrast to many other mesh generation procedures focusing on mesh quality, the aim of the present effort is to obtain the capability to robustly and with minimal user interference produce hybrid meshes suitable for engineering computations. The authors acknowledge that there are a multitude of challenging flow problems which will still require the use of different mesh generation tools and algorithms in order to create an acceptable computational mesh. Nevertheless, it is anticipated that a fast and comparatively robust tool will allow for significant savings in time and effort where meshes for many different routine flow simulations must be generated. An example of such applications may be the automated creation of a series of meshes for a full aircraft model in a windtunnel, which often requires a separate mesh for each experiment in order to correctly capture wall effects. Another use case which could possibly benefit substantially is the application to military aircraft with multiple external stores, where many different geometric configurations need to be handled with limited effort.

Generally speaking, the purpose of creating a hybrid prismatic mesh is to substantially increase mesh resolution in boundary layers, where high Reynolds number flows exhibits large gradients in the wall-normal direction. It is, however, important to note that the extent of the prismatic mesh layer does not necessarily correspond directly to the size of the boundary layer; in fact, the outer limit of the prismatic layer will usually far exceed the boundary layer displacement thickness. This is an intentional feature of the present method which permits a smooth volumetric transition between the uppermost prismatic elements and adjacent tetrahedra. Therefore, only the lower part of the prismatic layer (near the wall) is actually intended to resolve the physical boundary layer, which must be taken into account when considering the number of layers and the wall-normal element size expansion ratio.

3. Method

The mesh generation strategy is based on four phases, starting with the creation of a sufficiently resolved surface mesh. In a second step, the envelope mesh of the prismatic boundary layer mesh is determined; the robustness of this stage is the primary contribution of the present work. Thirdly, tetrahedral elements are generated to fill the volume between the envelope of the prismatic layer and the farfield boundaries, and finally, pentahedral elements are grown between adapted wall and envelope mesh.

3.1. Surface meshes

For the generation of unstructured surface meshes for full aircraft configurations, the open-source program SUMO[1] can be used. The geometry is in this case represented by a moderate number of parametric surfaces, which can be linear-cubic or bicubic polynomial spline surfaces, analytically defined, or non-uniform rational b-spline (NURBS) surfaces.

Triangular surface meshes are then constructed by a method which accounts for a three-dimensional version of the Delaunay criterion [15]. Two mesh refinement passes are used in order to fulfill a set of triangle quality criteria which can either be determined based on heuristics or specified by the user. Heuristics exploit information which is available due to the fact the the geometry modeling component included in SUMO is specialized for aircraft configurations.

If the use of SUMO is not desired, a triangular surface mesh created with any other mesh generation tool can be used. Such meshes can be imported from files in CGNS [16], SU2, legacy VTK, the NetCDF-based format used by the TAU code, or the widespread STL format. Spherical far-field boundaries can in this case be generated automatically from a small set of user-defined parameters.

3.2. Prismatic envelope

Once a surface mesh is available, the volume to be filled with prismatic elements is determined by constructing a second triangular surface with identical topology at a locally varying distance from the wall mesh. In the following text, this second triangulated surface is called the envelope, as it encloses the entire prismatic layer. In order to handle the multitude of geometric difficulties which can occur at this stage [3], the shell surface is not a simple extrusion of the surface mesh along local normals. Instead, a sequence of passes are applied as follows:

1. feature extraction and surface node classification;
2. selective smoothing of growth directions;
3. selective smoothing of layer height;
4. local untangling and warp reduction;
5. global collision avoidance;

with the aim of transforming the envelope surface into a suitable upper boundary of the prismatic mesh region.

Surface node classification. The first step in creating the envelope surface is to detect and classify nodes with respect to geometric properties which necessitate special treatment of either local height or growth direction. Local criteria, such as the angle between the normal vectors of adjacent triangles, are evaluated in order to assign a set of flags to each vertex of the wall mesh. Multiple flags may be combined; a vertex can, for instance, be classified as lying on a detected mesh feature line (ridge) and simultaneously carry a *convex* and *concave* flag, which would make it a local saddle point.

Some typical examples are vertices which are part of edges (which, again, may be blunt or sharp, convex or concave), or corner vertices. Figure 1 shows the wall surface of the F-16XL aircraft used in the cranked arrow wing project (CAWAPI,[17]), where dark regions mark vertices which carry any flag differing from the one used to indicate a locally flat surface. One of the factors controlling the classification is a user-provided feature angle. This angle defines a limit for the dihedral angle between triangles, above which edges are considered to be part of feature lines. Therefore, the feature angle should be chosen larger than the maximum dihedral angle occurring between wall triangles in smoothly curved regions. This value is typically known to the surface mesh generator which can therefore be exploited automatically.

Note that the feature detection phase only evaluates local geometry, and ignores information such as boundaries between components (mesh regions). This is seen at the junction between the canopy and fuselage or the vertical tail and root of the vertical tail in Figure 1.

[1] Available from http://www.larosterna.com/sumo.html

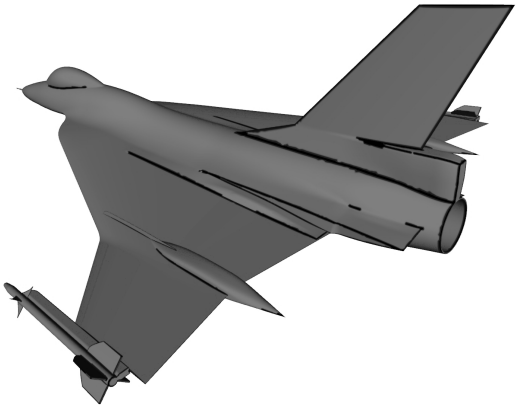

Fig. 1. Detected geometric features.

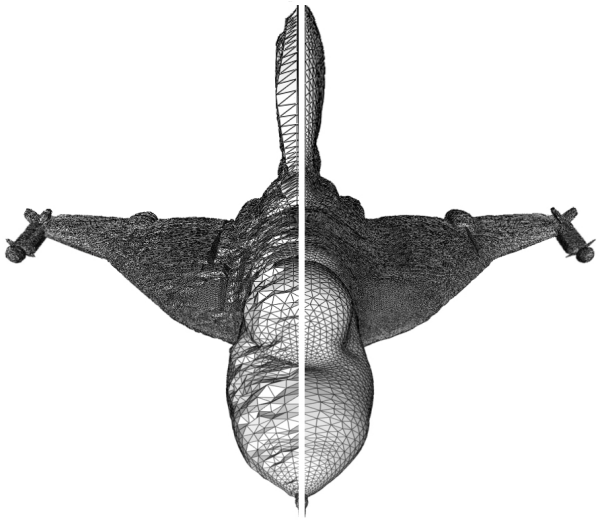

Fig. 2. Effect of smoothing on envelope surface.

Envelope smoothing. The envelope surface enclosing the prismatic layer is initialized using the vertex-based surface normal vectors obtained by angle-weighted averages of the normal vectors of coincident triangles at each vertex, according to the method by Thürmer and Wüthrich [18]. Furthermore, the local layer thickness is initialized by first computing the mean length l_j of the incident edges at vertex j. Then, the initial local layer height h_j is determined from a user-supplied first prism height h_0 according to

$$h_j = h_0 \frac{1 - r_j^n}{1 - r_j} \quad \text{with} \quad r_j = \min\left[\left(\frac{l_j}{h_0}\right)^{\frac{1}{n-1}}, r_m\right], \tag{1}$$

where r_j is the local height expansion ratio of adjacent prisms. This growth ratio is limited to a user-supplied maximum value of r_m. The height distribution is chosen such that the wall-normal edge length of the outermost pentahedron is as close as possible to the edge lengths of the abutting tetrahedron. The reason for this selection is that, for finite-volume methods employing a vertex-centered discretization, the accuracy of gradient reconstruction algorithms tends to improve when the lengths of edges connected to the same vertex do not vary too much [19].

Due to triangle size variations and considerable local deviation between normal directions, the initial envelope surface is often very irregular in shape. The left side of Figure 2 shows an example; no sound prismatic mesh can be generated between the wall and this boundary surface.

Therefore, three smoothing passes are performed either indirectly or directly on the envelope surface. During the first pass, a modified Laplacian smoothing operator is applied to the height field by means of a small number of Jacobi iterations, where the height modification operation is adapted depending on the value of the vertex flags identified earlier. Secondly, a similar procedure is utilized to smooth the growth directions (normals) with a different pattern of flag-dependent modifications. Modifications based on vertex flags aim at avoiding a deterioration of feature resolution which would result from isotropic smoothing; as an example, normal directions for vertices marked as being part of feature edges are smoothed exclusively with respect to other normals on the same edge.

On the right side of Figure 2, the resulting envelope surface after smoothing is displayed. This surface is sufficiently well-shaped to permit the generation of good-quality pentahedral cells between wall and envelope mesh.

Local untangling. The envelope surface generated in this way will often contain self-intersections which would result in entangled pentahedral elements. To remove such intersections, an algorithm using only local, edge-based geometry is run first, followed by a second, global procedure.

In the first phase, an edge-based limiter for the prismatic layer height is applied. For each edge $\boldsymbol{e}$ in the wall mesh, the angles β_1 and β_2 between the edge direction and the vertex normal vectors in the endpoints of the edge are

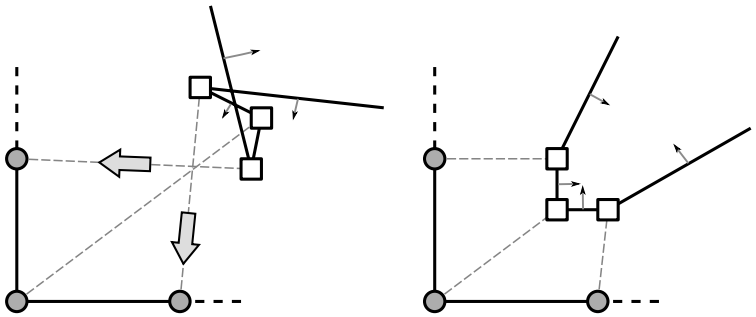

Fig. 3. Untangling and warp reduction.

computed. Using $\gamma = \pi - \beta_1 - \beta_2$, the maximum permitted height at the edge endpoint j is then obtained from

$$h_j < |\boldsymbol{e}| \frac{\sin \beta_j}{\sin \gamma} \quad \text{when} \quad 0 < \gamma < \pi. \tag{2}$$

Once the local height is reduced accordingly, a simple smoothing of the height variable in the ring-2 neighborhood of all modified vertices is performed in order to soften the transition.

Warp reduction. The initial prismatic layer can contain cells which are strongly warped or even inverted. A warped pentahedron is shaped such that the angle between a triangle on the envelope and the wall normal direction is small. In some cases, such as the one shown in the left part of Figure 3, envelope triangle normal vectors may even point toward the wall when the wall-normal sides of the pentahedron intersect. Note that, in this sketch, an unrealistically coarse surface mesh is displayed to improve clarity. In order to eliminate such degenerate cases, the local height at the affected vertices is reduced to the largest possible value which avoids warping. The permitted height is found by means of bisection such that a positive minimum pentahedral corner angle is obtained, as indicates by the grey arrows in Figure 3. Since the limit of zero local height always yields an acceptable pentahedron, a reduction of the local height will always converge to a permissible solution.

Encroaching bodies.. In some instances, the geometry of the envelope surface cannot be determined by purely local geometric considerations. An example is the region between a rear-mounted engine nacelle and the fuselage shown in Figure 4, where the unmodified prismatic region envelope enclosing the nacelle would intersect the layer around the fuselage.

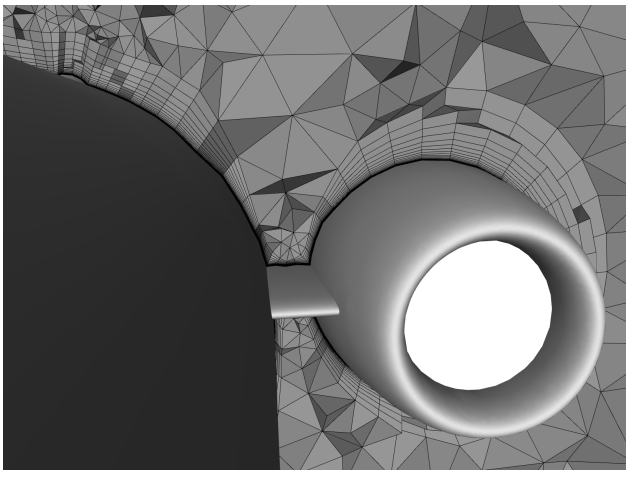

Fig. 4. Envelope retraction to avoid intersection.

As is visible in Figure 4, the potential collision was detected and the layers on both sides were retracted, i.e. their height reduced in order to alleviate the problem. Efficient detection of self-intersection is performed by means of a efficient search tree data structure, namely a balanced binary tree bounding volume hierarchy. This particular data structure is used to perform fast queries of the geometric neighbourhood of a point on the envelope to test for the presence of other points within a given critical radius. If any such point is then categorized as indicating collision by comparison of the corresponding local normal directions, the local envelope height is reduced accordingly. As the relation between height and intersection state can be rather irregular for complex geometry, the process of envelope modification and testing for self-intersection is repeated until no further collisions are detected. Often, just two or three iterations are required; for very intricate configurations, as many of 20 repetitions can be needed which still does not account for a significant amount of computational effort.

Most of the envelope modification procedures described process each vertex independently and could therefore be performed in parallel, should the need arise. At present however, the (serial) generation of the tetrahedral mesh region dominates expended computation time, which is why only some steps of the above envelope construction have been parallelized.

3.3. Tetrahedral mesh

Before prismatic elements are generated in the region between wall and envelope layer, tetrahedral elements are created in the space between envelope and farfield boundaries. The efficient tetrahedral mesh generation program TetGen[2] developed by Han Si [20,21] is employed for this purpose. A number of element quality parameter can be passed to TetGen in order to control the level of refinement. In many cases, these element quality criteria cannot be satisfied unless some of the boundary triangles are split to allow for smaller adjacent tetrahedra. As the method used to create prismatic elements builds upon the assumption that wall and envelope mesh have the exact same topology, such triangle splits need to be propagated to the wall mesh as well.

In order to relate vertices inserted by TetGen on the envelope to the wall surface, each boundary triangle passed to TetGen is tagged with an integer identifying the corresponding wall triangle. Newly created envelope triangles inherit the same tag, so that it is possible to split the wall mesh by inserting a new wall vertex at the barycentric coordinates of the envelope vertex created by TetGen. The lookup procedure used to determine whether a vertex received from TetGen corresponds to a previously defined envelope vertex or requires a wall triangle split makes use of the same search data structure also employed in resolving collisions between encroaching envelopes as explained in Section 3.2.

The use of a Delaunay procedure in TetGen entails certain restrictions on the type of surface mesh appropriate for the present approach. Triangle meshes dominated by large aspect ratio elements, such as, for example, surface meshes obtained by splitting a structured quadrilateral mesh with stretched cells, are not suitable, since they rarely permit a high-quality conforming Delaunay tetrahedralization in their vicinity. Although the envelope construction procedure of Section 3.2 is fully capable of handling such surface discretizations, the interface mesh seen by TetGen would still contain very many ill-shaped triangles. Fulfilling a tetrahedral quality criterion based on the three-dimensional Delaunay property then leads to extremely many splits and entirely unacceptable node counts as the stretched interface mesh is transformed into a more isotropic refinement. Therefore, surface triangles should be restricted to an aspect ratio of below about six.[3]

Most solver for high-speed flows require that the mesh is sufficiently fine in the vicinity of shocks. Insufficient mesh resolution leads to overly strong numerical dissipation which has the effect of smearing out the shock, thus resulting in inaccurately computed pressure distributions. Adaptive mesh refinement is known to be an effective way to mitigate this problem in a largely automated fashion; nevertheless, the additional steps to be performed by the user add to the (already large) complexity incurred by using CFD. Furthermore, aggressive solution-based mesh adaptation is best suited for problems with essentially steady, i.e. predominantly attached flow.

In order to obtain a reasonable resolution of shocks without mesh adaptation, the tetrahedral mesh generation phase can be split into two TetGen passes. In the first pass, a relatively low-quality background mesh is created quickly by specifying a relatively large permitted circumsphere radius-to-edge ratio of around 1.5. Then, this mesh is employed

[2] Available from http://www.tetgen.org

[3] Preliminary results with the latest version 1.5 of TetGen indicate that this restriction may be lifted.

to specify a nodal element size map which approximates the smooth element size transition which is typical of an advancing-front process. In a second pass, TetGen is run in refinement mode with the nodal element size map on the background mesh as an input. The resulting tetrahedral mesh is characterized by a gradual element size transition visible in Figures 5 - 7.

3.4. Extrusion of prismatic elements

Starting from the wall surface, prismatic elements are finally generated by filling the space between wall and envelope with a prescribed number of prismatic element layers. The height of the first prismatic cell is a user-defined value which is applied throughout the entire mesh. Starting from the second cell, a constant growth ratio is maintained such that the last cell reaches the local layer thickness. Hence, the growth ratio may differ substantially between regions of the mesh with different triangles sizes. However, the element generation process is extremely fast since the pentahedral layer is essentially structured in the wall-normal direction, meaning that the number of pentahedral layers is identical everywhere.

In some areas, the layer thickness may be so small that the above simple procedure would result in a growth rate below one, that is, the cell height would diminish away from the wall. An equally spaced distribution of cell heights across the layer is selected if such a situation arises. A typical geometrical feature exhibiting this pattern would be a detailed model of the narrow lateral gap between two control surfaces.

Mesh quality can be further improved by aligning the first few pentahedra (nearest to the wall) not with the previously determined smoothed growth direction, but starting with the actual wall normal vector and transitioning gradually. This leads to almost perfectly shaped pentahedra near the wall, but can lead to tangled elements when very sharp edges with acute ridge angles are present.

3.5. Limitations

With the current implementation, all triangular boundaries are treated in the same manner, that is, prismatic layers are generated everywhere. For this reason, special boundaries such as symmetry planes (where no prismatic layers are desired) can not easily be integrated, since such boundaries would require re-meshing to allow a connection of the coincident prismatic layer with matching quadrilateral elements on the symmetry plane.

Furthermore, some CAD models and derived surface meshes contain degenerate geometric details, which would not usually exist in an actual product but are caused by modeling simplifications. Such degeneracy can lead to the case where it is not possible to define a vertex normal which sustains an angle of less than 90° with the normals of all of the coincident triangles, which renders the construction of well-defined pentahedral elements impossible. In order to properly handle such geometric degenerate points, multiple vertex normal directions would need to be defined and the algorithm of Section 3.4 must be substantially modified.

4. Example applications

Figures 5-8 are intended to show a selection of example meshes generated with the present approach, ranging from a relatively benign jet trainer prototype to a simplified Airbus A320 undercarriage assembly. Figure 8 is the mesh quality histogram displayed by ICEM-CFD for the SUMO-generated mesh shown in Figure 7.

Unfortunately, purely geometrical element criteria appear to represent poor practical measures of overall mesh quality. Therefore, a comparative study was performed with two different configurations, where RANS solutions obtained from meshes generated by SUMO were compared with those computed on available grids with similar resolution created with other software systems. The aim of this study was to evaluate whether the quickly generated SUMO meshes yielded similar simulation results in sensitive metrics. In the absence of a more rigorous criterion relating mesh geometry to solution convergence and accuracy, this is regarded as a relevant investigation.

Two very different cases were selected for this comparison. The first is the Common Research Model (CRM) wing-body-tail geometry representative of a wide-body airliner, which should present a comparatively well-behaved case. In contrast, the second model used is the F-16XL research aircraft evaluated at a flight condition known to involve complex flow features.

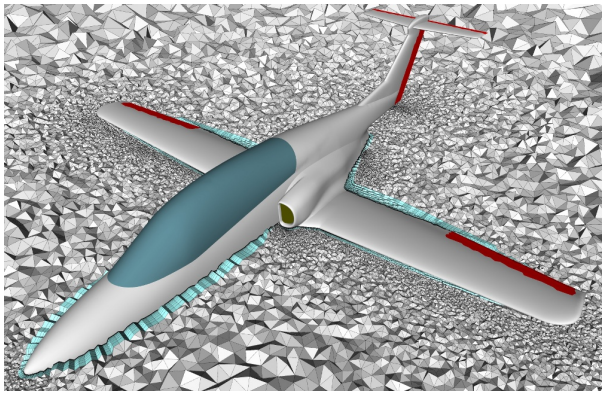

Fig. 5. Grid cut-plane around Ranger 2000 jet trainer prototype.

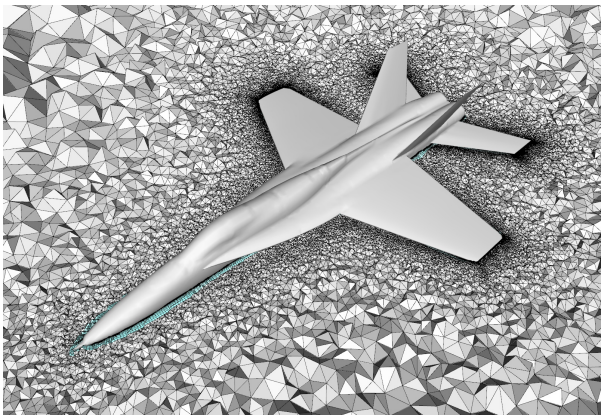

Fig. 6. Mesh around YF-17 fighter configuration.

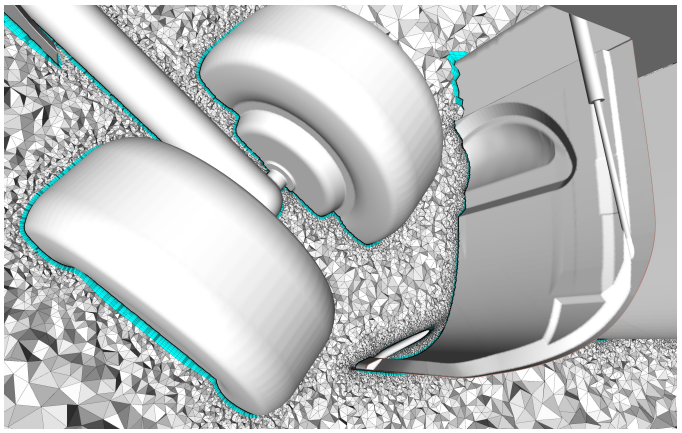

Fig. 7. Cut through mesh around A320 main landing gear and doors.

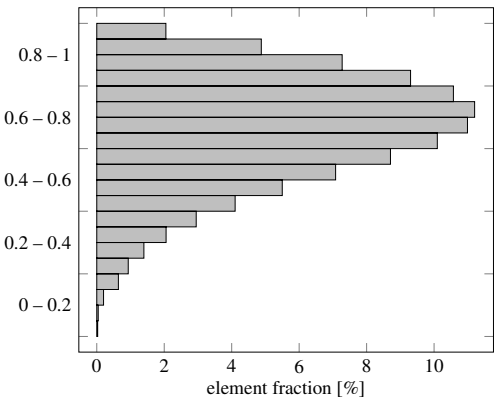

Fig. 8. Typical mesh quality histogram.

4.1. Common Research Model

The Common Research Model (CRM) is a geometrically fairly simple wing-body-stabilizer wind-tunnel configuration developed for applied CFD validation studies [22]. It is intended to reproduce many of the large-scale flow features present on commercial aircraft currently in service. In this study flow simulations for the CRM were performed for a set of Mach numbers between 0.7 and 0.87 and angles of attack from zero to six degrees at a chord Reynolds number of 5 million. The prismatic grids generated for this model follow the general grid guidelines from the fourth drag prediction workshop (DPW-4)[4]. In order to focus the evaluation on the quality of the volume mesh generation process, a surface mesh from the DPW-4 website was used, namely the mirrored medium-resolution grid generated by DLR, containing 1.2 million surface triangles.

4.2. F-16XL Research Aircraft

The F-16XL aircraft on the other hand is a comparatively complex configuration that presents many unique aerodynamic challenges across its aggregate flight envelope [17]. This vehicle incorporates cranked delta wings designed for efficient supersonic cruise. As a result, the wings are thin, highly swept, and feature relatively small leading-edge

[4] http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/Workshop4/gridding_guidelines_4.html

radii. The configuration investigated here also includes wing-tip missiles, air-dams, pods, narrow gaps and other geometrical details that tend to be a challenge for many prismatic grid generators [23].

In this study the current prismatic grid generator was used to produce a hybrid grid of the F-16XL aircraft for a flow simulation of the very challenging transonic Flight Condition 70 (FC70, [24]) at Mach 0.97 and an angle of attack of 4.3° and a chord-based Reynolds number of 89 million. At this angle of attack, significant interaction between the shock and a vortex emanating from the sharp inboard leading edge is expected to occur. At the time of writing, it has been found rather difficult to obtain satisfactory agreement of computational results with flight test data at this particular flight condition with multiple different mesh and solver combinations. As the focus of the present paper is the acceleration of the mesh generation process, no experimental results are shown here. Instead, pressure distributions computed with meshes originating from different generators are compared.

The surface grid used for evaluating mesh quality is the mirrored initial KTH/FOI surface grid [24] with 316 000 surface triangles. 35 prismatic layers were generated with an initial wall distance of 10^{-6} m ensuring that $y^+ < 1$ everywhere.

5. Simulation Results

The flow solver Edge, developed at the Swedish defence research establishment (FOI), was used for the comparative simulations [25]. Turbulence was modelled using the Spalart-Allmaras (SA) one equation model for the CRM cases and the Hellsten, Wallin and Johansson $k-\omega$ explicit algebraic Reynolds stress model (EARSM) for the F-16XL case. Since the purpose of the simulations was not to predict performance or investigate flow phenomena, but rather to evaluate the dependency of computed results on properties of the volume mesh generation scheme, only steady-state runs were performed. It is understood that this may well lead to inaccurate results for some cases.

Edge is based on a finite-volume formulation where a median dual grid forms the control volumes with the unknowns allocated in the vertices of the primary grid. The governing equations are integrated to steady state by means of a line-implicit approach in regions with highly stretched elements (e.g. the prismatic layer) and explicitly elsewhere with a three-stage Runge-Kutta time integration scheme. Convergence to steady state is accelerated by a full approximation multigrid scheme. Solid surfaces were associated with weak adiabatic wall boundary conditions, while weak characteristic exterior conditions were used for the far-field boundary. For the F-16XL case the engine inlet and mixing plane conditions where set to the same values as presented by Boelens et al. [23]. Details of the mesh element and node counts can be found in Table 2.

5.1. Transonic wind-tunnel model CRM

The initial mesh for the CRM geometry contained a comparatively coarse tetrahedral region with 7.7 million elements, generated with a tetrahedral quality criterion (circumsphere radius to edge length ratio [20]) of 1.2. For this mesh, some differences in pitch moment were observed when compared with the advancing-front mesh containing nearly twice as many elements in the tetrahedral domain. These differences could easily be attributed to a lack of shock resolution. Therefore, the SUMO-based mesh was re-generated with a tetrahedral quality criterion of 1.05, leading to a mesh with 17.6 million tetrahedral cells, compared with 15 million cells in the reference grid. The more restrictive tetrahedral shape requirements can only be obtained by allowing TETGEN to split the triangulated envelope surface where necessary. In the end, the splits are carried through to the wall surface, thus leading to a total of 1.4 million additional nodes in the prismatic layer. Note that these results were obtained before the two-pass approach to tetrahedral mesh generation described in Section 3.3 was implemented.

A consequence of the different algorithms applied for the tetrahedral mesh generation is that this region of the mesh is not directly comparable in terms of density distribution and element size transition. Additionally, the SUMO-generated mesh contains a full prismatic region with all 35 layers everywhere. In contrast, other programs such as TRITET can eliminate some of the layers where that is deemed advantageous. Mostly due to these two factors and a minor increase in tetrahedron density, the mesh generated by SUMO features about 27% more nodes despite being based on the same surface mesh.

Figure 9 shows the drag polar for the CRM model for mach numbers 0.7, 0.85 and 0.87, where the more finely resolved farfield mesh was used as explained above. A comparison at lift coefficient $C_L = 0.5$ indicates that the drag

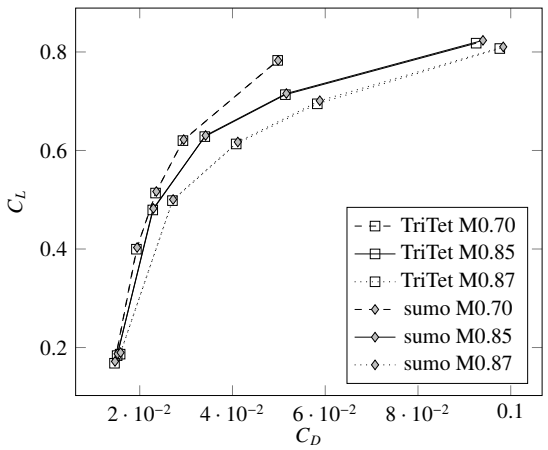

Fig. 9. CRM drag polar comparison.

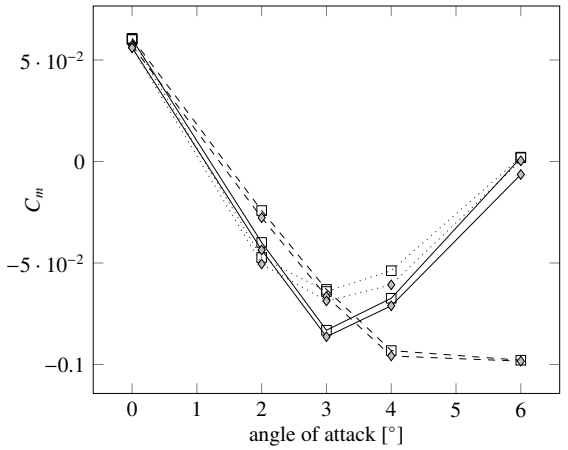

Fig. 10. CRM pitch moment coefficient comparison.

only differs by 0.04% between the SUMO-generated and reference grids. This difference is most likely significantly less than the overall accuracy of the simulation. In Figure 10 the pitch moment C_m as function of angle of attack is shown. Again good agreement between meshes is observed for moderate lift coefficients, while slightly larger differences occur at $\alpha = 4°$.

Figure 11 and Figure 12 show the surface pressure and skin friction distribution of the CRM at Mach 0.85 and angle of attack 2°. The upper half of each image shows the solution based on the TRITET-generated grid while the lower side shows the solution based on the SUMO-generated grid with 17.6 million tetrahedral elements. No significant discrepancies between the two solutions have been recognized.

5.2. F-16XL Flight Condition 70

Table 1 compares integrated forces and moments between the advancing-front mesh generated using TRITET and the hybrid prismatic/Delaunay grid produced using SUMO and TETGEN for flight test FC70 of the F-16XL case. While the discrepancies in lift force and pitching moment are minor, there is also a small difference in total drag and a larger

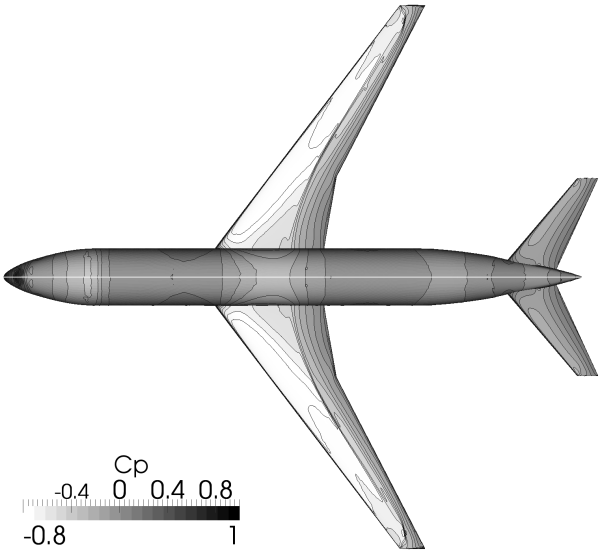

Fig. 11. Pressure comparison, bottom: SUMO-mesh, top: TRITET.

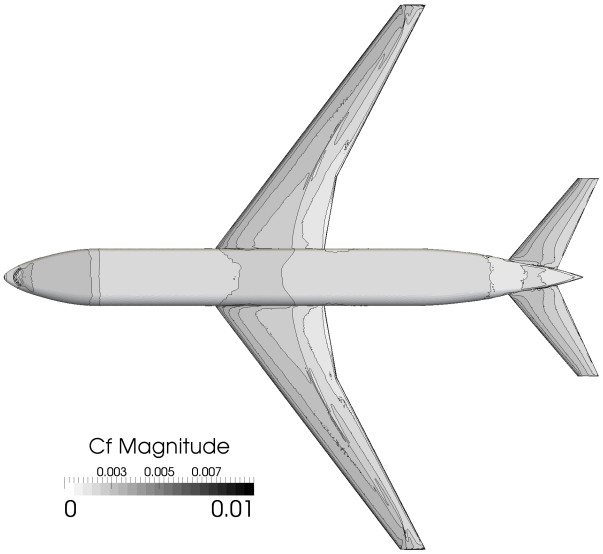

Fig. 12. Wall friction coefficient comparison.

Coefficient	TriTet	sumo	Difference
C_L	0.12010	0.12017	-0.06%
C_D	0.03760	0.03750	1.0 counts
C_{D_f}	0.00508	0.00449	5.9 counts
C_m	-0.12931	-0.12925	+0.05%

Table 1. F-16XL forces and moment coefficients at Mach 0.97, angle of attack 4.3° and chord Reynolds number 89 million.

difference in friction drag – about 6 drag counts. Figure 13 s hows the surface pressure distribution, where the upper half shows the reference solution (TriTet-mesh) while the lower half presents the results from the simulation on the sumo grid. The main observation is the slightly larger supersonic region over the canopy in the sumo grid solution. Otherwise the differences are small given the very complex flow topology.

Further studying the tetrahedral region (see Figure 14) in the near-field around the body, it is clear that the sumo produced grid (right) has a significantly larger growth ratio, however the near-field resolution is kept higher in the tetrahedral domain compared to the TriTet-mesh (left). Even in this case, the sumo-mesh was generated before the two-pass TetGen option was available.

5.3. Mesh generation performance

In Table 2, the size and computing time needed for the generation of the different hybrid are shown. Note that, depending on the specific geometry in question, the achievable degree of automation/scripting, and user experience, some of the tools listed may require non-negligible manual effort and repeated attempts, which is not considered here. Use of sumo entails only the setting of a small number of parameters, such as the desired height of the first cell and the number of layers.

While the method used in sumo has been outlined in Section 3, a short explanation is needed for the two other mesh generators mentioned in Table 2. The mesh generator TriTet [7] builds up the prismatic region layer by layer, whereupon the tetrahedral domain is gradually filled by means of a serial advancing front method. For the CRM case, which is typical in this respect, about 80% of the computing time expended by TriTet is used by the tetrahedral mesh generation phase.

The commercial software icem-cfd incorporates multiple algorithms for hybrid mesh generation. An advancing front method conceptually similar to the approach implemented in TriTet requires somewhat longer computing times than TriTet. The mesh listed in Table 2, however, was generated with another algorithm resembling the procedure

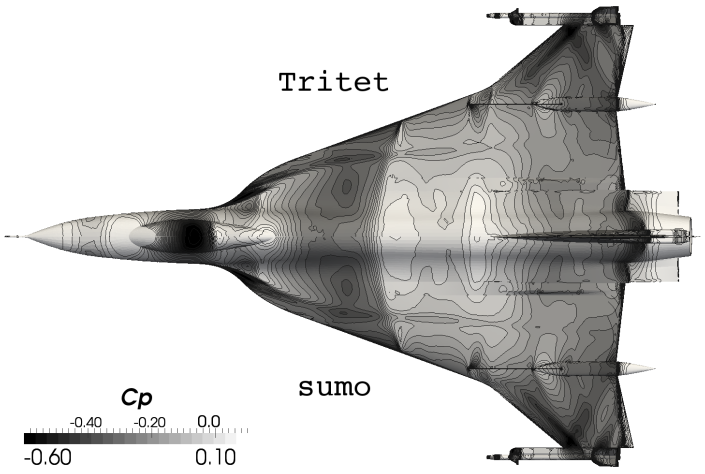

Fig. 13. Surface pressure comparison for F-16XL.

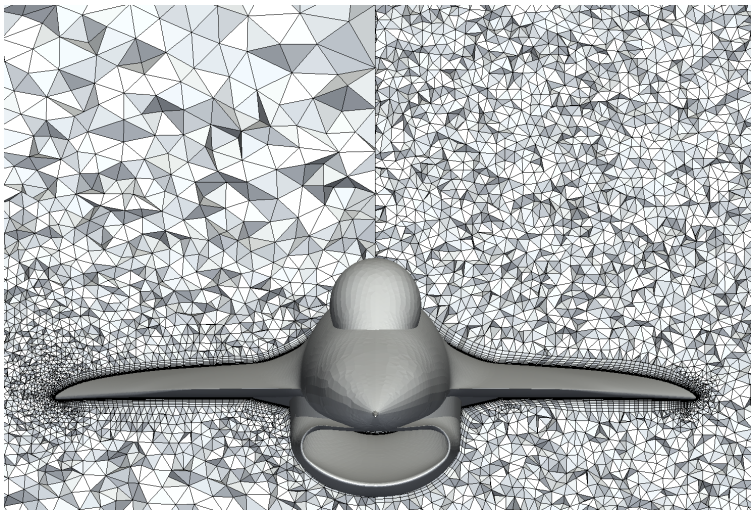

Fig. 14. Grid comparison of near-field tetrahedral resolution for F-16XL case

Grid	Generation time Prisms	[h:min:sec] Total	Surface triangles	Prism cells	Tet cells	Total cells	Total nodes
CRM TriTet	1:32:45	7:20:32	1212k	34.3M	15.1M	50.1M	20.3M
CRM icem-cfd	1:18:56	(1:51:23)	1212k	42.4M	14.2M	57.9M	23.7M
CRM sumo	0:02:01	0:10:17	1295k	45.4M	17.6M	62.9M	25.8M
F-16XL TriTet	0:17:30	4:50:21	316k	12.2M	14.8M	27.2M	8.8M
F-16XL sumo	0:01:01	0:04:38	316k	11.3M	16.3M	27.6M	8.3M

Table 2. Element counts and wall-clock generation times of the computational grids.

employed by sumo, where the prismatic region envelope is generated once and then split into pentahedral cells, while the surrounding volume is filled by means of a Delaunay tetrahedralization. This process is substantially faster at just below 2 hours, but does not result in a mesh which passes the pre-processor of the Edge flow solver due to tangled or inverted cells. Additional investigations into the controlling parameters are therefore needed before a fully representative comparison can be established.

For a resolution comparable to the TriTet mesh, i.e. 25.8 million nodes, sumo required a total computation time of slightly more than 10 minutes. A mesh for the F-16XL configuration with 8.3 million nodes was created in less than 5 minutes total time.

6. Conclusions

When comparing the mesh generation timings, an interesting observation can be made. For the common situation where a parallel CFD solver is run on a dedicated compute cluster of moderate size, the analyst may be evaluating post-processed results of a steady-flow simulation based on a sumo-generated mesh before an advancing front mesh has even been generated. This is a substantial advantage of the presented open-source method.

Obviously, this does not mean that there is no need for high-quality advancing-front mesh generation tools. A substantial proportion of relevant geometries and flight conditions likely require more detailed control over mesh generation parameters than what is available in a the present hybrid Delaunay implementation. However, for routine solutions where serial mesh generation time is a severe bottleneck, sumo or the underlying libraries can be used to accelerate the turnaround time considerably.

Unfortunately, the level of robustness originally aimed for has not yet been reached to the desired extent. There are still geometric configurations for which the envelope construction technique employed here fails to create an

outer layer surface of sufficient quality, even when this should be geometrically possible. The typical failure mode is then that the mesh fails to pass the solver pre-processing checks due to low prismatic element quality leading to unreasonable dual mesh properties.

Future developments should include the ability to re-mesh symmetry planes or other special boundary surfaces to match the adjacent pentahedral elements. This could be achieved with relative ease for the common case of perfectly plane boundary surfaces; a general solution of this particular problem does, however, require that a representation of the continuous surface geometry (i.e. not just a discrete surface mesh) of the specific boundaries is available at the volume mesh generation stage. While certainly possible, the present solution has so far attempted to avoid this requirement in order to allow the fast generation of hybrid prismatic grids based on already existing, or externally-generated, discrete surfaces meshes.

References

[1] T. J. Baker. Mesh Generation: Art or Science? *Progress in Aerospace Sciences*, 41:29–63, 2005. doi:10.1016/0898-1221(92)90044-I.

[2] F.T. Johnson, E.N. Tinoco, and N.J. Yu. Thirty years of development and application of CFD at Boeing Commercial Airplanes, Seattle. *Computers and Fluids*, 34(10):1115 – 1151, 2005. doi:10.1016/j.compfluid.2004.06.005.

[3] Y. Ito. Challenges in unstructured mesh generation for practical and efficient computational fluid dynamics simulations. *Computers & Fluids*, 85(1):47–52, October 2012. doi:10.1016/j.compfluid.2012.09.031.

[4] S. Crippa. Improvement of Unstructured Computational Fluid Dynamics Simulations Through Novel Mesh Generation Methodologies. *Journal of Aircraft*, 48(3):1036–1044, 2011. doi:10.2514/1.C031219.

[5] S. Z. Pirzadeh. Advanced Unstructured Grid Generation for Complex Aerodynamic Applications. *AIAA Journal*, 48(5):904–915, 2010. doi:10.2514/1.41355.

[6] ANSYS, Inc. *ANSYS ICEM CFD 12.1 USER MANUAL*, 2009. URL: http://www.ansys.com.

[7] L. Tysell, T. Berglind, and P. Eneroth. Adaptive Grid Generation for 3D Unstructured Grids. In *6th International Conference on Numerical Grid Generation in Computational Field Simulations*, June 1998.

[8] J. Vassberg, M. DeHaan, and T. Sclafani. Grid generation requirements for accurate drag predictions based on OVERFLOW calculations. In *16th AIAA Computational Fluid Dynamics Conference*, 2003. AIAA paper 2003-4124.

[9] H. Luo, S. Spiegel, and R. Löhner. Hybrid Grid Generation Method for Complex Geometries. *AIAA Journal*, 48(11):2639–2647, 2010. doi:10.2514/1.J050491.

[10] M. Svärd, J. Gong, and J. Nordström. An accuracy evaluation of unstructured node-centred finite volume methods. *Applied Numerical Mathematics*, 58(8):1142 – 1158, 2008. doi:10.1016/j.apnum.2007.05.002.

[11] T. Baker. Mesh Generation for the Computation of Flowfields over Complex Aerodynamic Shapes. *Computers Math. Applic.*, 24(5/6):103–127, 1992.

[12] R.V. Garimella and M.S. Shephard. Boundary layer mesh generation for viscous flow simulations. *International Journal for Numerical Methods in Engineering*, 49(1-2):193–218, 2000.

[13] Y. Ito and K. Nakahashi. Improvements in the reliability and quality of unstructured hybrid mesh generation. *International Journal for Numerical Methods in Fluids*, 45(1):79–108, 2004.

[14] M. Tomac and D. Eller. From Geometry to CFD Grids – An Automated Approach for Conceptual Design. *Progress in Aerospace Sciences*, 47(8):589–596, 2011. doi:10.1016/j.paerosci.2011.08.005.

[15] L. P. Chew. Guaranteed-Quality Mesh Generation for Curved Surfaces. In *Proceedings of the Ninth Annual Symposium on Computational Geometry*, San Diego, May 1993.

[16] D. Poirier, S. Allmaras, D. McCarthy, M. Smith, and F. Enomoto. The CGNS System. In *29th AIAA Fluid Dynamics Conference*. AIAA, June 1998. AIAA Paper 98-3007.

[17] C. J. Obara and J. E. Lamar. Overview of the Cranked-Arrow Wing Aerodynamics Project International. *Journal of Aircraft*, 46(2):355–368, 2009. doi:10.2514/1.34957.

[18] G. Thürmer and C. A. Wüthrich. Computing vertex normals from polygonal facets. *J. Graph. Tools*, 3(1):43–46, 1998.

[19] J.F. Thompson, B.K. Soni, and N.P. Weatherill. *Handbook of Grid Generation*. Taylor & Francis, 1998.

[20] H. Si and K. Gaertner. Meshing Piecewise Linear Complexes by Constrained Delaunay Tetrahedralizations. In *Proceedings of the 14th International Meshing Roundtable*, pages 147–163, San Diego, September 2005.

[21] H. Si. On Refinement of Constrained Delaunay Tetrahedralizations. In *Proceedings of the 15th International Meshing Roundtable*, September 2006.

[22] J. Vassberg, M. DeHaan, M. Rivers, and R. Wahls. Development of a Common Research Model for Applied CFD Validation Studies. In *26th AIAA Applied Aerodynamics Conference*, 2008. doi:10.2514/6.2008-6919.

[23] O. J. Boelens, K. J. Badcock, S. Gortz, S. Morton, W. Fritz, S. L. Jr. Karman, T. Michal, and J. E. Lamar. Description of the F-16XL Geometry and Computational Grids Used in CAWAP. *Journal of Aircraft*, 46(2):369–376, 2009.

[24] A. Rizzi, A. Jirásek, J. E. Lamar, S. Crippa, K. J. Badcock, and O. J. Boelens. Lessons Learned from Numerical Simulations of the F-16XL Aircraft at Flight Conditions. *Journal of Aircraft*, 46(2):423–441, 2009. doi:10.2514/1.35698.

[25] P. Eliasson. Edge, a Navier-Stokes solver for unstructured grids. In *Finite Volumes for Complex Applications III*, pages 527–534, June 2002.